

AMENDMENTS TO THE CLAIMS

Claims 1-5, 7-19 and 21-66 were pending at the time of the Action.

Claims 11-14, 25-28 and 44-47 are canceled in this Response.

Claims 1, 3-5, 7-9, 15, 17-19, 21-23, 29-31, 34-36, 39-41, 48-50, 53-55, 58-60, 63 and 65 are amended in this Response.

Claims 1, 7-9, 15, 21-23, 63 and 65 are independent claims.

Accordingly, claims 1-5, 7-10, 15-19, 21-24, 29-43 and 48-66 remain pending.

Claim 1 (currently amended): A method for creating a binary tree data structure, the data structure embodied in a computer-readable medium, from an ordered list of at least four elements, each element having an associated value in the list, comprising:

determining whether the list has an even or odd number of elements;

separating the list into left side groupings and right side groupings based on whether the list has an even or odd number of elements, the groupings being separated by a parent node defined by a median of the list, wherein the median is a left element of two middle values of the list when the list has an even number of elements, or the median is a middle value element of the list when the list has an odd does not have an even number of elements;

inserting-creating left side descendent nodes into-of the binary tree by successively finding a median of each left side grouping and linking each found median to the previous median;

inserting-creating right side descendent nodes into-of the binary tree by successively finding a median of each right side grouping and linking each found median to the previous median;

wherein when a grouping has an even number of elements, the median is a left element of two middle values of the grouping;

1 wherein when a grouping ~~does not have an even~~ has an odd number of
2 number of elements, the median is a middle value element of the grouping; and
3 wherein the elements of the lists include logged events.

4 Claim 2 (original): A computer-readable medium having stored thereon
5 computer-executable instructions for performing the method of claim 1.

6
7 Claim 3 (currently amended): The method of claim 1, wherein each
8 element in the list includes a pointer to a corresponding node of a plurality of
9 nodes in a partially assembled binary tree, wherein each node has a left child
10 pointer, and wherein ~~inserting~~ creating the left side nodes farther comprises
11 assigning a value to the left child pointer of at least one of the nodes.

12 Claim 4 (currently amended): The method of claim 1; wherein each
13 element in the list includes a pointer to a corresponding node of a plurality of
14 nodes in a partially assembled binary tree, wherein each node has a right child
15 pointer, and wherein ~~inserting~~ creating the right side nodes further comprises
16 assigning a value to the right child pointer of at least one of the nodes.

17 Claim 5 (currently amended): The method of claim 1, wherein ~~inserting~~
18 creating the left side descendent nodes comprises inserting the left side descendent
19 nodes into a partially assembled version of the binary tree, wherein ~~inserting~~
20 creating the right side descendent nodes comprises ~~inserting~~ creating the right side
21 descendent nodes into the partially assembled version of the binary tree, and
22 wherein the list is a linked list that acts as a wrapper around the partially
23 assembled version of the binary tree.

24 Claim 6 (canceled).
25

1 Claim 7 (currently amended): A method for creating a binary tree data
2 structure, the data structure embodied in a computer-readable medium, from an
3 ordered list of at least four elements, each element having an associated value in
4 the list, comprising:

5 determining whether the list has an even or odd number of elements;

6 separating the list into left side groupings and right side groupings based
7 on whether the list has an even or odd number of elements, the groupings being
8 groupings separated by a parent node defined by a median of the list, wherein the
9 median is a left element of two middle values of the list when the list has an even
10 number of elements, or the median is a middle value element of the list when list
11 has an odd does not have an even number of elements;

12 ~~inserting creating~~ left side descendent nodes ~~into of~~ the binary tree by
13 successively finding a median of each left side grouping and linking each found
14 median to the previous median;

15 ~~inserting creating~~ right side descendent nodes ~~into of~~ the binary tree by
16 successively finding a median of each right side grouping and linking each found
17 median to the previous median;

18 wherein when a grouping has an even number of elements, the median is a
19 left element of two middle values of the grouping;

20 wherein when a grouping ~~does not have an even~~ has an odd number of
21 number of elements, the median is a middle value element of the grouping; and

22 wherein the elements of the list include data representing number of times
23 one or more threads of execution have passed through one or more code modules.
24

25 Claim 8 (currently amended): A method for creating a binary tree data
structure, the data structure embodied in a computer-readable medium, from an
ordered list of at least four elements, each element having an associated value in
the list, comprising:

determining whether the list has an even or odd number of elements;

1 separating the list into left side groupings and right side groupings based on
2 whether the list has an even or odd number of elements, the groupings being
3 separated by a parent node defined by a median of the list, wherein the median is a
4 left element of two middle values of the list when the list has an even number of
5 elements, or the median is a middle value element of the list when the list has an
6 odd does not have an even number of elements;

7 inserting-creating left side descendent nodes into-of the binary tree by
8 successively finding a median of each left side grouping and linking each found
9 median to the previous median;

10 inserting-creating right side descendent nodes into-of the binary tree by
11 successively finding a median of each right side grouping and linking each found
12 median to the previous median;

13 wherein when a grouping has an even number of elements, the median is a
14 left element of two middle values of the grouping;

15 wherein when a grouping ~~does not have an even~~ has an odd number of
16 number of elements, the median is a middle value element of the grouping; and

17 wherein the inserted-created right and left descendant nodes include data
18 representing a number of times one or more threads of execution have passed
19 through one or more code modules.

20 Claim 9 (currently amended): A method for creating a binary tree data
21 structure, the data structure embodied in a computer-readable medium, from an
22 ordered list of at least four elements, each element having an associated value in
23 the list, comprising:

24 separating the list into left side groupings and right side groupings based on
25 whether the list has an even or odd number of elements, the groupings being
separated by a parent node defined by a median of the list, wherein the median is a
left element of two middle values of the list when the list has an even number of
elements, or the median is a middle value element of the list when the list has an

~~odd does not have an even~~ number of elements;

~~inserting-creating~~ left side descendent nodes ~~into~~ of the binary tree by successively finding a median of each left side grouping and linking each found median to the previous median;

~~inserting-creating~~ right side descendent nodes ~~into~~ of the binary tree by successively finding a median of each right side grouping and linking each found median to the previous median;

wherein when a grouping has an even number of elements, the median is a left element of two middle values of the grouping;

wherein when a grouping ~~does not have an even~~ has an odd number of number of elements, the median is a middle value element of the grouping; and

wherein the ~~inserted-created~~ right and left descendant nodes include one or more pointers to data representing a number of times one or more threads of execution have passed through one or more code modules.

Claim 10 (original): The method of claim 1, wherein the list is an ordered linked list.

Claim 11 (canceled)

Claim 12 (canceled)

Claim 13 (canceled)

Claim 14 (canceled)

Claim 15 (currently amended): A method for creating a binary tree data structure, the data structure embodied in a computer-readable medium, from an ordered list of at least four elements, each element having an associated value in

the list, comprising:

determining whether the list has an even or odd number of elements;

separating the list into left side groupings and right side groupings based on whether the list has an even or odd number of elements, the groupings being separated by a parent node defined by a median of the list, wherein the median is a left element of two middle values of the list when the list has an even number of elements, or the median is a middle value element of the list when the list has an odd does not have an even number of elements;

~~inserting~~ creating right side descendent nodes ~~into~~ of the binary tree by successively finding a median of each right side grouping and linking each found median to the previous median;

~~inserting~~ creating left side descendent nodes ~~into~~ of the binary tree by successively finding a median of each left side grouping and linking each found median to the previous median;

wherein when a grouping has an even number of elements, the median is a right element of two middle values of the grouping;

wherein when a grouping ~~does not have an even~~ has an odd number of number of elements, the median is a middle value element of the grouping; and

wherein the elements of the list include logged events.

Claim 16 (original): A computer-readable medium having stored thereon computer-executable instructions for performing the method of claim 15.

Claim 17 (currently amended): The method of claim 15, wherein each element in the list includes a pointer to a corresponding node of a plurality of nodes in a partially assembled binary tree, wherein each node has a right child pointer, and wherein ~~inserting~~ creating the right side nodes further comprises assigning a value to the right child pointer of at least one of the nodes.

1 Claim 18 (currently amended): The method of claim 15, wherein each
2 element in the list includes a pointer to a corresponding node of a plurality of
3 nodes in a partially assembled binary tree, wherein each node has a left child
4 pointer, and wherein ~~inserting-creating~~ the left side nodes further comprises
5 assigning a value to the left child pointer of at least one of the nodes.

6 Claim 19 (currently amended): The method of claim 15, wherein ~~inserting~~
7 ~~creating~~ the right side descendent nodes comprises ~~inserting-creating~~ the right side
8 descendent nodes into a partially assembled version of the binary tree, wherein
9 ~~inserting-creating~~ the left side descendent nodes comprises ~~inserting-creating~~ the
10 left side descendent nodes into the partially assembled version of the binary tree,
11 and wherein the list is a linked list that acts as a wrapper around the partially
12 assembled version of the binary tree.

13 Claim 20 (canceled).

14 Claim 21 (currently amended): A method for creating a binary tree data
15 structure, the data structure embodied in a computer-readable medium, from an
16 ordered list of at least four elements, each element having an associated value in
17 the list, comprising:

18 determining whether the list has an even or odd number of elements;

19 separating the list into left side groupings and right side groupings based on
20 whether the list has an even or odd number of elements, the groupings being
21 separated by a parent node defined by a median of the list, wherein the median is a
22 left element of two middle values of the list when the list has an even number of
23 elements, or the median is a middle value element of the list when the list has an
24 odd does not have an even number of elements;

25 ~~inserting-creating~~ right side descendent nodes ~~into of~~ the binary tree by
successively finding a median of each right side grouping and linking each found

1 median to the previous median;

2 ~~inserting-creating~~ left side descendent nodes ~~into-of~~ the binary tree by
3 successively finding a median of each left side grouping and linking each found
4 median to the previous median;

5 wherein when a grouping has an even number of elements, the median is a
6 right element of two middle values of the grouping;

7 wherein when a grouping ~~does not have an even~~ has an odd number of
8 number of elements, the median is a middle value element of the grouping; and

9 wherein the elements of the list include data representing a number of times
10 one or more threads of execution have passed through one or more code modules.

11 Claim 22 (currently amended): A method for creating a binary tree data
12 structure, the data structure embodied in a computer-readable medium, from an
13 ordered list of at least four elements, each element having an associated value in
14 the list, comprising:

15 determining whether the list has an even or odd number of elements;

16 separating the list into left side groupings and right side groupings based on
17 whether the list has an even or odd number of elements, the groupings being
18 separated by a parent node defined by a median of the list, wherein the median is a
19 left element of two middle values of the list when the list has an even number of
20 elements, or the median is a middle value element of the list when the list has an
21 odd ~~does not have an even~~ number of elements;

22 ~~inserting-creating~~ right side descendent nodes ~~into-of~~ the binary tree by
23 successively finding a median of each right side grouping and linking each found
24 median to the previous median;

25 ~~inserting-creating~~ left side descendent nodes ~~into-of~~ the binary tree by
successively finding a median of each left side grouping and linking each found
median to the previous median;

wherein when a grouping has an even number of elements, the median is a

1 right element of two middle values of the grouping;

2 wherein when a grouping ~~does not have an even~~ has an odd number of
3 number of elements, the median is a middle value element of the grouping; and

4 wherein the ~~inserted-created~~ right and left descendant nodes include data
5 representing a number of times one or more threads of execution have passed
6 through one or more code modules.

7 Claim 23 (currently amended): A method for creating a binary tree data
8 structure, the data structure embodied in a computer-readable medium, from an
9 ordered list of at least four elements, each element having an associated value in
10 the list, comprising:

11 determining whether the list has an even or odd number of elements;

12 separating the list into left side groupings and right side groupings based on
13 whether the list has an even or odd number of elements, the groupings being
14 separated by a parent node defined by a median of the list, wherein the median is a
15 left element of two middle values of the list when the list has an even number of
16 elements, or the median is a middle value element of the list when the list has an
17 odd ~~does not have an even~~ number of elements;

18 ~~inserting-creating~~ right side descendent nodes ~~into-of~~ the binary tree by
19 successively finding a median of each right side grouping and linking each found
20 median to the previous median;

21 ~~inserting-creating~~ left side descendent nodes ~~into-of~~ the binary tree by
22 successively finding a median of each left side grouping and linking each found
23 median to the previous median;

24 wherein when a grouping has an even number of elements, the median is a
25 right element of two middle values of the grouping;

26 wherein when a grouping ~~does not have an even~~ has an odd number of
27 number of elements, the median is a middle value element of the grouping; and

28 wherein the ~~inserted-created~~ right and left descendant nodes include one or

1 more pointers to data representing a number of times one or more threads of
2 execution have passed through one or more code modules.

3 Claim 24 (original): The method of claim 15, wherein the list is an ordered
4 linked list.

5
6 Claim 25 (canceled)

7
8 Claim 26 (canceled)

9
10 Claim 27 (canceled)

11
12 Claim 28 (canceled).

13 Claim 29 (currently amended): The method of claim 7, wherein each
14 element in the list includes a pointer to a corresponding node of a plurality of
15 nodes in a partially assembled binary tree, wherein each node has a left child
16 pointer, and wherein ~~inserting~~ creating the left side nodes further comprises
17 assigning a value to the left child pointer of at least one of the nodes.

18 Claim 30 (currently amended): The method of claim 7 wherein each
19 element in the list includes a pointer to a corresponding node of a plurality of
20 nodes in a partially assembled binary tree, wherein each node has a right child
21 pointer, and wherein ~~inserting~~ creating the right side nodes further comprises
22 assigning a value to the right child pointer of at least one of the nodes.

23 Claim 31 (currently amended): The method of claim 7, wherein ~~inserting~~
24 creating the left side descendent nodes comprises ~~inserting~~ creating the left side
25 descendent nodes into a partially assembled version of the binary tree, wherein

1 ~~inserting-creating~~ the right side descendent nodes comprises ~~inserting-creating~~ the
2 right side descendent nodes into the partially assembled version of the binary tree,
3 and wherein the list is a linked list that acts as a wrapper around the partially
4 assembled version of the binary tree.

5 Claim 32 (previously presented): The method of claim 7, wherein the list is
6 an ordered linked list.

7 Claim 33 (previously presented): A computer-readable medium having
8 stored thereon computer-executable instructions for performing the method of
9 claim 7.

10 Claim 34 (currently amended): The method of claim 8, wherein each
11 element in the list includes a pointer to a corresponding node of a plurality of
12 nodes in a partially assembled binary tree, wherein each node has a left child
13 pointer, and wherein ~~inserting-creating~~ the left side nodes further comprises
14 assigning a value to the left child pointer of at least one of the nodes.

15 Claim 35 (currently amended): The method of claim 8, wherein each
16 element in the list includes a pointer to a corresponding node of a plurality of
17 nodes in a partially assembled binary tree, wherein each node has a right child
18 pointer, and wherein ~~inserting-creating~~ the right side nodes further comprises
19 assigning a value to the right child pointer of at least one of the nodes.

20 Claim 36 (currently amended): The method of claim 8, wherein ~~inserting~~
21 ~~creating~~ left side descendent nodes comprises inserting the left side descendent
22 nodes into a partially assembled version of the binary tree, wherein ~~inserting~~
23 ~~creating~~ the right side descendent nodes comprises ~~inserting-creating~~ the right side
24 descendent nodes into the partially assembled version of the binary tree, and
25

1 wherein the list is a linked list that acts as a wrapper around the partially
2 assembled version of the binary tree.

3 Claim 37 (previously presented): The method of claim 8, wherein the list is
4 an ordered linked list.

5
6 Claim 38 (previously presented): A computer-readable medium having
7 stored thereon computer-executable instructions for performing the method of
8 claim 8.

9 Claim 39 (currently amended): The method of claim 9, wherein each
10 element in the list includes a pointer to a corresponding node of a plurality of
11 nodes in a partially assembled binary tree, wherein each node has a left child
12 pointer, and wherein ~~inserting-creating~~ the left side nodes further comprises
13 assigning a value to the left child pointer of at least one of the nodes.

14 Claim 40 (currently amended): The method of claim 9, wherein each
15 element in the list includes a pointer to a corresponding node of a plurality of
16 nodes in a partially assembled binary tree, wherein each node has a right child
17 pointer, and wherein ~~inserting-creating~~ the right side nodes further comprises
18 assigning a value to the right child pointer of at least one of the nodes.

19
20 Claim 41 (currently amended): The method of claim 9, wherein ~~inserting~~
21 ~~creating~~ the left side descendent nodes comprises ~~inserting-creating~~ the left side
22 descendent nodes into a partially assembled version of the binary tree, wherein
23 ~~inserting-creating~~ the right side descendent nodes comprises ~~inserting-creating~~ the
24 right side descendent nodes into the partially assembled version of the binary tree,
25 mad wherein the list is a linked list that acts as a wrapper around the partially
assembled version of the binary tree.

1 Claim 42 (previously presented): The method of claim 9, wherein the list is
2 an ordered linked list.

3
4 Claim 43 (previously presented): A computer-readable medium having
5 stored thereon computer-executable instructions for performing the method of
6 claim 9.

7 Claim 44 (canceled)

8
9 Claim 45 (canceled)

10
11 Claim 46 (canceled)

12
13 Claim 47 (canceled)

14 Claim 48 (currently amended): The method of claim 21, wherein each
15 element in the list includes a pointer to a corresponding node of a plurality of
16 nodes in a partially assembled binary tree, wherein each node has a right child
17 pointer, and wherein ~~inserting~~creating the right side nodes further comprises
18 assigning a value to the right child pointer of at least one of the nodes..

19 Claim 49 (currently amended): The method of claim 21, wherein each
20 element in the list includes a pointer to a corresponding node of a plurality of
21 nodes in a partially assembled binary tree, wherein each node has a left child
22 pointer, and wherein ~~inserting~~creating the left side nodes further comprises
23 assigning a value to the left child pointer of at least one of the nodes.

24
25 Claim 50 (currently amended): The method of claim 21, wherein ~~inserting~~

1 creating the right side descendent nodes comprises inserting-creating the right side
2 descendent nodes into a partially assembled version of the binary tree, wherein
3 inserting-creating the left side descendent nodes comprises inserting-creating the
4 left side descendent nodes into the partially assembled version of the binary tree,
5 and wherein the list is a linked list that acts as a wrapper around the partially
6 assembled version of the binary tree.

7 Claim 51 (previously presented): The method of claim 21, wherein the list
8 is an ordered linked list.

9 Claim 52 (previously presented): A computer-readable medium having
10 stored thereon computer-executable instructions for performing the method of
11 claim 21.

12 Claim 53 (currently amended): The method of claim 22, wherein each
13 element in the list includes a pointer to a corresponding node of a plurality of
14 nodes in a partially assembled binary tree, wherein each node has a right child
15 pointer, and wherein inserting-creating the right side nodes further comprises
16 assigning a value to the right child pointer of at least one of the nodes.

17 Claim 54 (currently amended): The method of claim 22, wherein each
18 element in the list includes a pointer to a corresponding node of a plurality of
19 nodes in a partially assembled binary tree, wherein each node has a left child
20 pointer, and wherein inserting-creating the left side nodes further comprises
21 assigning a value to the left child pointer of at least one of the nodes.

22 Claim 55 (currently amended): The method of claim 22, wherein inserting
23 creating the right side descendent nodes comprises inserting-creating the right side
24 descendent nodes into a partially assembled version of the binary tree, wherein
25

1 ~~inserting-creating~~ the left side descendent nodes comprises ~~inserting-creating~~ the
2 left side descendent nodes into the partially assembled version of the binary tree,
3 and wherein the list is a linked list that acts as a wrapper around the partially
4 assembled version of the binary tree.

5 Claim 56 (previously presented): The method of claim 22, wherein the list
6 is an ordered linked list.

7 Claim 57 (previously presented): The computer-readable medium having
8 stored thereon computer-executable instructions for performing the method of
9 claim 22.

10 Claim 58 (currently amended): The method of claim 23, wherein each
11 element in the list includes a pointer to a corresponding node of a plurality of
12 nodes in a partially assembled binary tree, wherein each node has a right child
13 pointer, and wherein ~~inserting-creating~~ the right side nodes further comprises
14 assigning a value to the right child pointer of at least one of the nodes.

15 Claim 59 (currently amended): The method of claim 23, wherein each
16 element in the list includes a pointer to a corresponding node of a plurality of
17 nodes in a partially assembled binary tree, wherein each node has a left child
18 pointer, and wherein ~~inserting-creating~~ the left side nodes further comprises
19 assigning a value to the left child pointer of at least one of the nodes.

20 Claim 60 (currently amended): The method of claim 23, wherein ~~inserting~~
21 ~~creating~~ the right side descendent nodes comprises ~~inserting-creating~~ the right side
22 descendent nodes into a partially assembled version of the binary tree, wherein
23 ~~inserting-creating~~ the left side descendent nodes comprises ~~inserting-creating~~ the
24 left side descendent nodes into the partially assembled version of the binary tree,
25

1 and wherein the list is a linked list that acts as a wrapper around the partially
2 assembled version of the binary tree.

3 Claim 61 (previously presented): The method of claim 23, wherein the list
4 is an ordered linked list.

5
6 Claim 62 (previously presented): A computer-readable medium having
7 stored thereon computer-executable instructions for performing the method of
8 claim 23.

9 Claim 63 (currently amended): A method for creating a binary tree data
10 structure, the data structure embodied in a computer-readable medium, from an
11 ordered list of at least four elements, each element having an associated value in
12 the list, comprising:

13 (a) determining whether the list has an even or odd number of elements;

14 (b) designating a median element of the list as a parent element based on
15 whether the list has an even or odd number of elements, wherein the parent
16 element divides the list into left side groupings and right side groupings, wherein
17 the median is a right element of two middle values of the list when the list has an
18 even number of elements, or the median is a middle value element of the list when
19 the list has an odd does not have an even number of elements;

20 (c) successively subdividing the right side groupings of the list and linking
21 each successive median element with a previous median element, thereby creating
22 right side descendent nodes in the binary tree;

23 (d) once each right side grouping has been exhausted as a result of step (c),
24 stepping back up the tree through each successive ancestor node until reaching an
25 element having left side groupings in the list, and, upon reaching an element
having a left side grouping in the list, proceeding to step (e);

(e) subdividing the left side groupings and linking a median element of the

1 left side grouping with the element reached in step (d), thereby creating a left side
2 descendent of the binary tree;

3 (f) if the left side descendent of step (e) has a right side grouping in the list,
4 repeating step (c) for the right side grouping;

5 (g) if the left side descendent of step (e) has no right side groupings, but has
6 a left side grouping, repeating step (e) for the left side grouping;

7 wherein the median element of a grouping is a right element of two middle
8 values of the grouping when the grouping has an even number of elements, or the
9 median is a middle value element of the grouping when the list ~~does not have an~~
10 even has an odd number of number of elements; and

11 wherein the elements of the list include data representing number of times
12 one or more threads of execution have passed through one or more code modules.

13 Claim 64 (previously presented): A computer-readable medium having
14 stored thereon computer-executable instructions for performing the method of
15 claim 63.

16 Claim 65 (currently amended): A method for creating a binary tree data
17 structure, the data structure embodied in a computer-readable medium, from an
18 ordered list of at least four elements, each element having an associated value in
19 the list, comprising:

20 (a) determining whether the list has an even or odd number of elements;

21 (b) designating a median element of the list as a parent element based on
22 whether the list has an even or odd number of elements, wherein the parent
23 element divides the list into left side groupings and right side groupings, wherein
24 the median is a right element of two middle values of the list when the list has an
25 even number of elements, or the median is a middle value element of the list when
the list has an odd ~~does not have an even~~ number of elements;

(c) determining if there are elements to the right of the parent element;

1 (d) if there are no elements to the right of the parent element, proceeding to
2 step (h);

3 (e) for the elements that are to the right of the parent element, finding a
4 median element;

5 (f) linking the median element of step (e) to the parent element so that the
6 median element is a child of the parent element;

7 (g) repeating steps (d) and (e), wherein the child element of step (f) is now
8 treated as the parent element in steps (d) and (e);

9 (h) locating a next element up on the tree that has elements to the left of it
10 and treating the element as a parent element in step (i);

11 (i) finding a median element of the elements to the left of the parent
12 element from step (h);

13 (j) linking the median element of step (i) to the parent element of step (h),
14 wherein the median element is a child of the parent;

15 (k) repeating steps (d) through (g), wherein the child element of step (j) is
16 treated as the parent element in step (d);

17 wherein the median element of a grouping is a right element of two middle
18 values of the grouping when the grouping has an even number of elements, or the
19 median is a middle value element of the grouping when the list ~~does not have an~~
20 even has an odd number of number of elements; and

21 wherein the elements of the list include data representing number of times
22 one or more threads of execution have passed through one or more code modules.

23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

Claim 66 (previously presented): A computer-readable medium having
stored thereon computer-executable instructions for performing the method of
claim 65.